AN EXAMPLE OF UNSUPERVISED NETWORKS KOHONEN'S SELF-ORGANIZING FEATURE MAP

Dagmar Niebur, Member, IEEE

Jet Propulsion Laboratory,
California Institute of Technology
PaswJenajCA91109, USA

Abstract- **Kohonen's self-organizing feature map belongs to a class of unsupervised artificial neural network commonly referred to as topographic maps. It serves two purposes, the quantization and dimensionality reduction of data. A short description of its history and its biological context is given. We show that the inherent classification properties of the feature map make it a suitable candidate for solving the classification task in power system areas like load forecasting, fault diagnosis and security assessment.**

1. B1OLOGICAJ> MOTIVATION

Topographically organized neural maps, like those which inspired Kohonen to develop his self-organizing feature map algorithm, have been observed in various parts of the central nervous system,

Cells in primary sensory cortex (that part of cortex which receives direct sensory input) can be characterized by their *receptive fields*. A receptive field of a cell is that part of the sensory world within which an adequate stimulus causes an excitatory or inhibitory response of the cell in question. For instance, cells in primary visual cortex are excited selectively by input to a small part of the visual field of the animal, and cells in the somato-sensory cortex respond preferentiall y to a sensation felt on a particular part of the skin.

Cortical neurons arc not arranged randomly but rather in functional areas. At a smaller scale, within the primary sensory and motor areas, cells are organized in ordered spatial maps which preserve the topography of the input space to some extent (*"topographic maps"*). For example neurons in a particular area of the somato-sensory cortex will correspond to the sensory input of neighboring fingers of the same had.

Starling already in the early seventies [von der Malsburg, 1973; Willshaw and von der Malsburg, 1976], there have been numerous attempts to understand the structure and the formation of such cortical maps by formal modeling While their model requires the postulated existence of so-called marker substances which guide the topographically ordered projection, more recent developmental models are based on cm-related electrical activity of input to the cortex. Linsker [1988] proposed a Inrrlti-layered network consisting of linear units in order to model orientation selective cells. His work was subsequently revised by Miller et al., [1989] for ocular dominance columns and orientation columns [Miller, 1994] and explained in terms of principal component analysis. Niebur and Wörgötter, [1994], however, have shown that such maps can also be described in terms of *very* simple geometrical constraints.

As discussed in [Ritter, 1988] for somatosensory maps and [Obermayer, 1993] for ocular dominance and orientation columns, the self-organizing feature map introduced in [Kohonen, 1982] provides an elegant and comparatively simple qualitative model which explains all these features with one mechanism.

2. A COMP[JTATIONAL MODEL FOR TOPOGRAPHIC MAPS

Kohonen [1982] proposed a formal model for the formation and function of topographic maps which he called "topology preserving map" and which is now known as *Kohonen's self-organizing feature map*. For a set of input signals, the map is designed to achieve the following tasks [Kohonen, 1989]:

1) Vector quantization of the input set,
2) Dimensionalit y reduction of the input space,
3) Preservation of the topological order present in the similarity relations of the input vectors

In the following sections its laterally connected architecture, winner-take-all processing, and unsupervised learning algorithm and the resulting properties are discussed.

2.1. *Architecture - lateral feedback through neighborhood relations*

The self-organizing feature map is an array of m processing elements (neurons) arranged on a lattice of arbitrary dimension. Most applications use a two-dimensional lattice but models where neurons are arranged on a (one-dimensional) line or in higher-dimensional spaces can be defined. For a given network, the input vectors x have a fixed dimension n. The n components of the input vectors are connected to each neuron in the lattice. A synaptic weight w_{ij} is defined for a connection from tbe jth component of the input vector to the ith neuron. Therefore, an n-dimensional vector $\mathbf{w}_i$ of synaptic weights is associated with each neuron i.

A neighborhood relationship is specified between the neurons of the Kohonen network. in the biological cortex, the connectivity of neurons decreases with their relative distance

In the computational model, this is behavior reproduced by introducing interactions between neurons whose strength decreases with their distance.

Fig. 1 shows a 4x4 Kohonen network which maps 3-dimensional input vectors to a two-dimensional map containing 16 neurons. Only neurons linked by a black line are connected, and only the weights belonging to neurons O and 8 are represented in this figure.

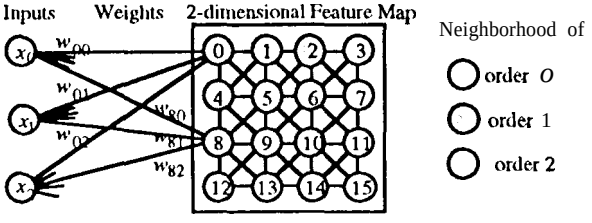

Figure 1: Neural network architecture of the self-organizing feature map. Assuming the most stimulated neuron is neuron 5, dark and lightly shaded neurons identify its first and second-order neighbors

For the genera] case of m neurons arranged on a two-dimensional lattice of length m l and width m_w, the connectivity is defined by the following neighborhood relation: With each neuron k (k = O, ... m-1) is associated its two-dimensional coordinate r(k) := (k_i, k_j), i = O, ... m_l-1 , j = O, ... row-l. The distance between neurons k and neuron 1 is then defined as a function of the indices i and j of k and l,

$$\mathrm{dist}(k, 1) = \|\mathrm{r}(k) - \mathrm{r}(1)\| = \mathrm{floor}(\|(k_i, k_j) - (l_i, l_j)\|), \quad \text{O)}$$

where floor(h) denotes the largest integer less or equal 10 h. With the Euclidean distance II II defined in (2)

$$\left\|\left(k_i, k_j\right) - \left(l_i, l_j\right)\right\| := \sqrt{\left(k_i - l_i\right)^2 + \left(k_j - l_j\right)^2} \qquad (2)$$

every neuron in Fig. 1 has at most 8 neighbors of order one for a distance < 2.

The neighborhood relationship can be chosen in an arbitrary way, and common ways to connect neurons are the association of four, six or eight neighbors for each neuron. In order to keep the subsequent figures simple, wc often omit the diagonal connections and the circles denoting the neurons.

Io principle, the number of neurons is independent of the dimension of the input vector and of the size of the training set. However, a small number of neurons can form only a small number of clusters, each one representing a large set of input vectors, which leads to a coarse discrimination of features in the input vectors. Depending on the application, such a coarse discrimination may not present a sufficiently detailed classification for a large training set and a larger number of neurons may be required.

In biological systems, the lateral connections between neurons implement excitatory and inhibitory links. In his original approach, Kohonen [1982] proposed a fully laterally connected network with distance-related strengths of synapses. Neurons close to each other on the grid have a positive (excitatory) coupling, whereas more distant neurons are coupled by negati ve (inhibitory) connections.

Plotting the functional influence between neurons as a function of the distance between them therefore yields the familiar *Mexican Hat Function, an* example of which is being shown in Fig. 2

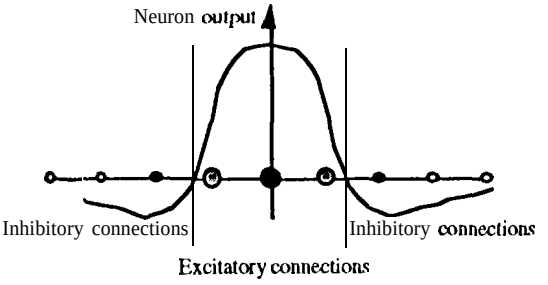

Figure 2: The *Mexican Hat Function:* a model for lateral excitation and inhibition of neurons on a one-dimensional net,

Kohonen [1982, 1989] showed that a network with this neighborhood function classifies input states without the need for an error signal (which is required with supervised learning techniques) and called it therefore the "Self-Organizing Feature Map." He also showed that the self-organization can be obtained without using the full neighborhood function shown in Fig. 2 but that a simpler, computationally more efficient neighborhood function is sufficient. It is, in fact, possible to omit the inhibitory connections, choosing a neighborhood function A(i, i^*, t) of exponential or Gaussian form, as in

$$\Lambda(i, i^*, t) = \exp\left(\frac{-\|r(i) - r(i^*)\|^k}{2\sigma(t)^2}\right), \qquad k = 1 \text{ or } 2 \qquad (3)$$

where $\sigma(t) = t^{-\beta}$, O < $\beta \leq$ 1 and t defines the (usually) discrete iteration time. The coordinates of neuron ion the two-dimensional grid are denoted as r(i), i* is the maximally excited neuron and i are the neighboring neurons. This function is shown schematically in Fig. 3.

Neurons which are connected to each ofher by strong excitatory connections form a functionally related neighborhood whose size is given by the parameter $\sigma(t)$ in (3). This parameter is conventionally called the "neighborhood size" of the neighborhood relation defined in (3). It is chosen as a decreasing function of iteration time.

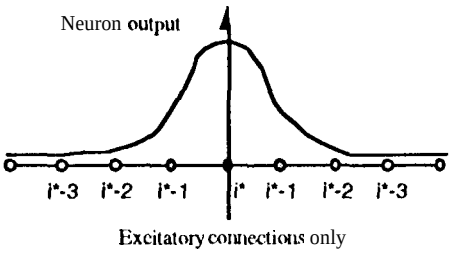

Figure 3: *The Gaussian Function:* a model for lateral excitation of neurons on a one-dimensional net.

3.2. The *self-organization algorithm*

2,2.1. *Processing - classification through competition*

When Kohonen introduced the network that bears his name, the algorithm he used for the processing of the input signal was similar to that of the logic threshold unit (see tutorial chapter on ANN concepts) and other biologically inspired artificial neural networks:

$$y_i = g(v_i) \quad \text{and} \quad v_i = \sum_{j=1}^{n} w_{ij}\, x_j = \langle w_i, x \rangle \tag{4}$$

In (4), *n is the* dimension of the input vectors and the gain function $g(v_i)$ is the winner-take-all function, i. *e.*

$$y_i = g(v_i) = \begin{cases} 1 & \text{for } i = i^* \\ 0 & \text{elsewhere} \end{cases}, \quad i = 1, \ldots, m \tag{5}$$

antt neuron i^* is selected such that its weight vector $\mathbf{w}_{i^*}$ is the most similar to the input vector. Similarity is defined by the angle between input and weight and measured by the scalar product. Thus

$$v_{i^*} = \max\{\langle \mathbf{w}_i, \mathbf{x} \rangle \text{ Ii} = 1, \ldots, /)1\} \tag{6}$$

where m is the number of neurons. Assuming for the moment as many output classes as neurons, neuron i^* respectively output yi* is called the *winner* of the competition.

Unsing the parallelogram equation it can be shown that for normalized inputs and weights, $\langle \mathbf{w}_i, \mathbf{x} \rangle$ takes its maximum for weight vector w i* if $\|\mathbf{w}_i - \mathbf{x}\|$ takes its minimum for $\mathbf{w}_i = \mathbf{w}_{i^*}$. As before, II II denotes the Euclidean distance. For normalized weight vectors (6) above and (7) below select the same winner, neuron i*. We can therefore select neuron i^* such that

$$v_{i^*} = \min\{\|\mathbf{w}_i - \mathbf{x}\| \; i = 1, \ldots, m\} \tag{7}$$

The two concepts of similarity for the selection of a winner are therefore eq u i valent for normalized vectors. However, the normalization is onl y needed in order to model the same type of neuron as the logic threshold unit. The theory of Kohonen networks does not require the normalization of input and weight vectors.

2.2.2. *Training - prototype generation through unsupervised learning*

As was mentioned previously, there is no need to specify the desired output in advance, when using the Kohonen algorithm and the training is therefore called *unsupervised* or *self-organized. The* algorithm is shown in schematic form in Fig. 4. Let *m* be the number of neurons with weight vectors $\mathbf{w}_k \in \Re^n, 1 \le k \le m$ anti $X = \{\mathbf{x} \in \Re^n | \mathbf{x}$ training vector) the training set. At each step *t* of the *learning* phase, a vector x, drawn randomly from the training set *X*, is presented as input to the network.

The neuron i* whose weight vector is closest (o the input vector x in the sense of the Euclidean distance, is selected. The weight vector wi* of this neuron is then adapted, becoming closer to the input vector according to the adaptation rule given in (8) below. The weight vectors of the neighboring neurons are also changed, by an amount which decreases with increasing distance to the winning weight vector. More precisely, at time *t*+1, the component *j* of weight vector wi is modified by adding

$$\begin{aligned} \Delta w_{ij} &= w_{ij}(t+1) - w_{ij}(t) \\ &= \eta(t)\, \Lambda(i, i^*, t)\, (x_j - w_{ij}(t)) \end{aligned} \tag{8}$$

The neighborhood function A of the adaptation rule, defined in 3, and the learning rate η have to be chosen such that the weight vectors converge to an equilibrium after a sufficientl y large number of input vectors from the training set X have been presented. This requires that the learning rate $\eta(t)$ decays with time.

Theoretical results concerning convergence and stability of the algorithm by Ritter and Schulten, [1988], suggest the following choice for $\eta(t)$

$$\eta(t) = t^{-\alpha} \quad \text{with} \quad 0 < \alpha \le 1 \tag{9}$$

and A(i, i^*,*t*) as defined in (3), see [Erwin *et al.*, 1992a, b]. A summary of the algorithm is listed in the following Fig. 4.

Since input vector x is drawn randomly from the training set containing a finite number of vectors (see step 2 in Fig. 4), any input vector may be chosen several times,

The c–criterion (step 6 in Fig. 4) is very time consuming for large networks and high-dimensional input spaces. Usually this criterion is therefore not applied and the simulation is instead halted after a pre-determined, large number of iterations. Furthermore, since A(i, i^*,*t*) decays exponentially with the distance between neurons, the change of the weights of neurons far away from the winner is negligible, Therefore A(i, i^*,*t*) is truncated, and only neurons

whose distance to the winner is smaller than the truncation threshold are updated in step 4.

After training, input vectors which are close (in the sense of the Euclidean distance) in the input space will stimulate neurons which are close to each other on the lattice. Some neurons may not be stimulated by any input vector. Grouping together all neurons stimulated by the same group of input vectors leads to the concept of neuron clusters representing classes of input vectors. Obviously, the number of such clusters can be smaller than *m*,

1) t :=0: initialize wij randomly for i = 1, . . ., m, j= 1, . . ., n,

2) Choose input vector $\mathbf{x} \in X$ randomly in the training set

3) Determine the neuron i^* such that its weight vector $\mathbf{w}_{i^*}$ is the closest to the input vector

$$\|\mathbf{w}_{i^*}(t) - \mathbf{x}\| = \min \|\mathbf{w}_i(t) - \mathbf{x}\| \quad \text{for all } i$$

4) Update the weight vectors $\mathbf{w}_i$, $i = 1, ..., m$:

$$\mathbf{w}_i(t+1) := \mathbf{w}_i(t) + \eta(t)\, A(i, i^*, t)\, (x - \mathbf{w}_i(t))$$

5) Increment the time $t := t+1$

5) if for several time intervals $\sqrt{\sum_{i=1}^{m} |\Delta w_{ij}|^2} > E$ then go to 2 else STOP

Figure 4: Training algorithm of the self-organizing feature map.

3.3.3. Self-organization as Hebbian learning

Kohonen's learning rule (8) obeys Hebb's postulated generalized learning principle. For convenience let us repeat this principle introduced in the previous tutorial chapter:

For the weight vector w of an artificial neuron, given input vector x and output y, synaptic learning can be expressed as the changes in the synaptic strength. These changes depend on the *learning factor* correlating output and input and the *forgetting factor* correlating output and weight:

$$\Delta \mathbf{w} = \eta\, y\, \mathbf{x} - \alpha\, y\, \mathbf{w} = \alpha\, y\, ((q/a)x - w) \qquad \text{(lo)}$$

with $\eta, a > 0$

With the choice of equal learning and forgetting rates $a = \eta$, and a winner-take-atl unit with neighborhood zero, the neuron output y will bc either zero or one, and Hebb's rule is satisfied. The neighborhood function however can be interpreted as white noise, which corrupts the output signal with normally distributed probability.

3.3.4. Self-organization by lateral feedback

We now discuss briefly alternatives to the update rule, step 3 in Fig, 4. The Kohonen algorithm requires that the distance between the input vector and all the weight vectors is determined at every iteration step. While this leads to a very efficient classification algorithm (see below), the implementation of this comparison in the biological brain, *i. e.* biological "hardware", may not be simple. The problematic step is the comparison of the outputs of all units for the selection of the maximum, This is a non-local process which requires information of the state of all units, to be monitored by a "master" unit.

Therefore, alternative updating schemes have been developed which do not require the explicit selection of a "winner" (vector with smallest distance to the input vector). In such schemes, the most responsive neuron is determined by a dynamical process leading to the formation of "activity bubbles" which converge into stable focused patches of activated neurons [Kohonen, 1989; Lehmann, 1993; Sirosh and Miikulainen, *in press*] An electronic implementation of this algorithm in analog VLSI technology has been developed by Vittoz et *al*, [1989].

3. CLASSIFICATION EXAMPLES OF RANDOMLY DISTRIBUTED INPUT VECTORS

The following section shows an illustration of the classification features of the Kohonen network. In the special cases where the input vector has the dimension 2 or 3, it is easy to represent this classification graphically, see also [Kohonen, 1989].

The feature map illustrated in Figs. 5, 6, and 7 consists of 10x 10 neurons. Each neuron is placed at the coordinates defined by the first and second component of its two-dimensionat weight vector. We assume that the input vectors arc uniformly distributed on the unit square,

Fig. 5 shows the initialization of the weight vectors which are randomly distributed in the input square but which do not exhibit any organizational structure. Neighboring neurons are randomly located on the square. Fig. 6 shows the state of the network after 200 input vectors randomly drawn from the unit square with uniform probability have been presented to the net work.

The map in Fig. 6 is already somewhat organized: neighboring neurons on the grid are usually located close to each other in the input space. But the organization is not perfect. The learning parameters are such that the neighborhood order is still 2. If in the next step, the input vector x is situated at the coordinates marked by the black circle, the winning neuron marked by the shaded circuit and all its neighbors of order one marked by the white circle and two (not marked in this figure 5) are updated simultaneously. Their coordinates will change in direction to the x.

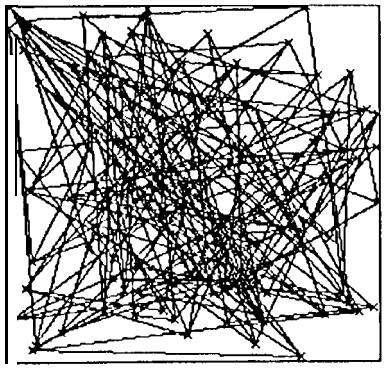

Fig. 5 Initial state of a 10x10 Kohonen network representing uniformly distributed two-dimensional input vectors. In the initial state, weight vector components are randomly distributed and the networks is unorganized.

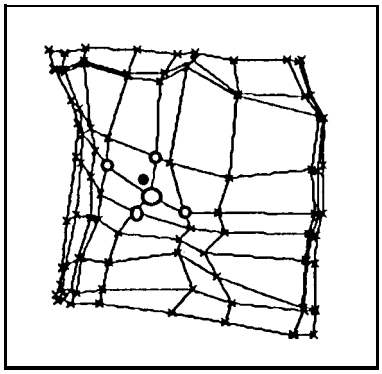

Fig. 6 organization of the Kohonen network after 200 training steps. The black dot denotes position of input at step 201. The shaded neuron will be the winner of the selection, the while circle denote direct neighbors .

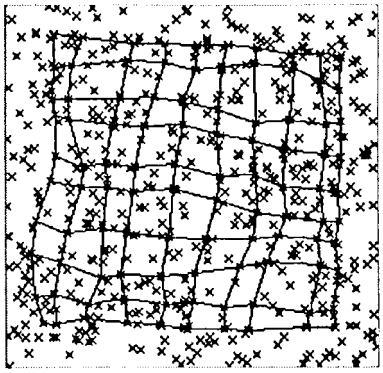

Fig. 7 Organization of the Kohonen network after 1000 training steps. The weight vectors are equally distributed in the input space. Their distribution reflects the uniform distribution of the input vectors shown as shaded marks

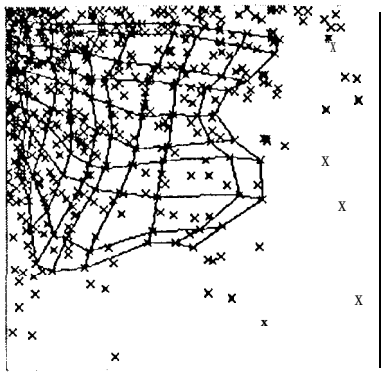

Fig. 8 Organization after 1000 integrations of a 10xI O Kohonen network representing Gaussian-distributed two-dimensional input vectors of means (O, 1). The weight vectors are densely distributed at the means. Their distribution reflects the Gaussian distribution of the input vectors shown as shaded marks

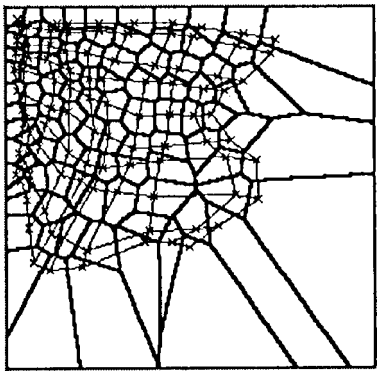

Fig. 9 Tessellation of the input space by the Kohonen network. Not e, that densely populated areas are represented by a larger number of neurons with a smaller class size.

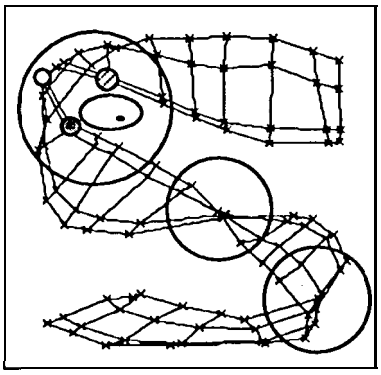

Fig. 10 Organization of a 4x30 Kohonen network with a b yperbol i call y decreasing neighborhood, representing' uniformly distributed two-dimensional input vectors. after 1000 training steps. White circles highlight areas where the map is distorted. Shaded circles denote neurons which violate the topological ordering principle. The ellipse shows one region of the input space where the network presents a distorted mapping of the input space.

Fig. 5- 10 were generated using an implementation of tfre Kohonen network developer at the Laboratoire de Microélectronique of the Ecole Polytechnique Fédérale de Lausanne [Demartines, 1991].

The change is proportional to their Euclidean distance to x, to the degree of their neighborhood to the winner and the learning rate.

A more regular pattern is obtained for larger numbers of training steps and a further decreasing neighborhood function and learning rate, Fig. 7 shows the evolution after 1000 training steps when the network has converged to a stationary state. The neighborhood order has been already decreased to zero, and in the last 300 steps only the winner was updated. The neurons are by now well organized and almost uniformly distributed in the input space, thus representing the same distribution as the input vectors. Two input vectors close to

each other in the input space will be classified by either the same neuron or by two neighboring neurons.

Fig. 8 represents the organization of the map for Gaussian distributed input vectors and Fig. 9 represents an approximation of the Voronoi tessellation obtained with the Kohonen map. Note that neighboring areas are represented by neighboring neurons.

An interesting example is shown in Fig, 10. Instead of a quadratic grid, neurons are placed on a rectangular grid. The size of the neighborhood function A further decreases with 1/t, instead of exponential] y. We can observe two phenomena.

First, the network assumes the form of a Peano curve. Its weight vectors still represent a quantization according to the distribution of the input vectors. Second, the topological representation of the input space is distorted in areas denoted by white circles. For example the dark shaded and the light shaded neuron arc direct neighbors on the grid. However neighboring input vectors drawn from the dotted region will most likely be classified by either the dark shaded neuron or a fourth order neighbor shown by a striped circle instead of its direct neighbor,

For the one-dimensional case, Erwin *et* al., [1991] have shown that for a Gaussian neighborhood function these topological defects can not occur.

An example with a higher-dimensional input space is shown in Fig. 11.

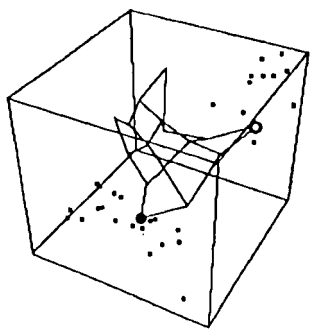

Figure 11: 4x4 Kohonen network representing uniformly distributed three-dimensional input vectors,

A two-dimensional 4x4 Kohonen network was trained with a total of 500 three-dimensional input vectors uniformly distributed over the volume of a three-dimensional unit cube.

The three-dimensional weight vectors should be distributed regularly in the cube. Due to the small size of the network, however, the comer neurons classify the extreme cases of vectors drawn from near the edges of the cube, although these input vectors may not be close in the sense of the Euclidean distance. This is illustrated at the example of two neurons (marked by a black and a white circle, respectively), The input vectors classified by these neurons are shown as black dots and it can be seen that they form two clouds adjacent to the two comers next to the corresponding neuron.

4. STATISTICAL PROPERTIES OF THE SELF-ORGANIZING FEATURE MAP

At a given time t, the state of the feature map consisting of *m* neurons is defined by the weight vectors $w_1, w_2, \ldots, w_m \in \Re^n$. Note that most of the results listed below are proven only for $n = 1$,

Let us assume that the input vectors $x \in \Re^n$ are distributed in the input space according to a given probability distribution P(x), In the adaptation rule defined by (8), the value of weight vector $w(t+1)$ at time $t+1$ depends on the weight vector w(1) at time t and on the input vector x presented at time $t+1$. Since x is drawn randomly from the distribution P(x), the state W of the feature map evolves stochastically in time, Since the probability of the state at time $t+1$ depends only on the state at time *t, the* state is a Markov process. For discrete time steps, the sequence $W(t)$ of feature map states forms a Markov chain.

Two questions have to be studied in the context of feature maps, first the convergence of the algorithm towards a stable state, second the properties of the map with respect to the preservation of the topology of the input space.

The first question, namely under which conditions for a probability distribution P(x), a learning rate $\eta(t)$ and a neighborhood function A(i, i^*,*t*) the Markov chain will converge to a stable equilibrium point, was addressed by several researchers.

In the case of a one-dimensionat real input space, Cottrell and Fort (1986) showed that the Markov chain converges almost surely to a unique stable equilibrium state if

x is uniformly distributed in $\Re^n$

and

$$A(i, i^*, t) := \begin{cases} 1 & \text{for } |i - i^*| < 2 \\ 0 & \text{elsewhere} \end{cases} \qquad (11)$$

and the learning rate fulfills the so-called "Robbins-Monroe" conditions:

$$\sum_{t=0}^{\infty} \eta(t) = +\infty \qquad \text{and} \qquad \sum_{t=0}^{\infty} \eta(t)^2 < \infty \qquad (12)$$

The conditions for convergence for general continuous distributions were weakened by Ritter [1988] who showed that the following condition on the learning rate is sufficient

$$\int_0^\infty \eta(t)dt = +\infty \qquad \text{and} \qquad \eta(t)\to 0 \qquad \text{for } t\to\infty \quad (13)$$

Note that the learning rate $\eta(t)$ defined in (9) fulfills these conditions. For the one-dimensional case Ritter [1991] showed further that the probability distribution of the weight vectors equals $P(x)^{2/3\ -\gamma(m)}$ where y(m) goes to zero for large nets, that is, for $m\to\infty$. '

Introducing a modification of the adaptation step, Yang and Dillon [1992] proved the convergence of the modified algorithm for the two-dimensional case.

For higher-dimensional input spaces and continuous probability distributions, Erwin et *al.* [1992a, b] proved the following negative result: There is no energy function whose minima corresponds to the stationary states of the system, because in this case the forces acting on the weights are not conservative. 'his means that in contrast to Hopfield nets, the stationary state cannot be found through optimization of a cost function. Energy functions can only be defined locally for each individual neuron [Tolat, *1990].*

in the case of *n*-dimensional *discrete* probability distributions, however, Ritter [1988] demonstrated the existence of a non-differentiable energy function bounded from below. This function may have several local minima. As in the case of simulated annealing techniques, the slow stochastic updating of the weight vectors, *i.* e. the slow decrease of the learning rate, facilitates the escape from these local minima. Ruzicka [1993] gives conditions on the learning rate and the neighborhood function for which the feature map converges to a stationary point. This is an important result for technical applications where the training set is often finite and its probability distribution is therefore always discrete. As a consequence, the convergence of the algorithm is guaranteed.

The second question to be discussed is the preservation of topologically ordered stales. In the one-dimensional case, if we number three neurons i, *j,* and *k* according to their position on the "lattice," the triple of corresponding weights $w_i, w_j,$ and w_k is called topological y ordered if

$$|w_i - w_j| < |w_i - w_k| \quad \text{for} \quad |i - j| < |i - k| \quad 04)$$

for $i, j, k = 1, \ldots, m$.

Cottrell and Fort [1986] showed that the topologically ordered state of the chain is absorbing. In other words, once the triple of weights is ordered, this topological order of w_i and w_j will not be destroyed. This result was extended by [Erwin *et* al., 1992a, b] for one-dimensional monotonically decreasing convex neighborhood functions as, e. g., the Gaussian function and the choice of $k = 2$ in (3). They showed that for such neighborhood functions, the topologically ordered states are the only stationary states of the system and that furthermore, there may exist meta-stable states for non-convex neighborhood functions. These meta-stable states do not respect the topological order of the map. An example for this type of behavior was illustrated in Fig. 10.

There is no straightforward definition of distortion in higher dimensions which would correspond to (14) in the one-dimensional case. A heuristic distortion measure was introduced by Bauer and Pawelzik [19921 who then showed experimentally that the distortion error can be reduced by increasing the dimension of the neuron lattice. However, one of the major advantages of the self-organizing feature map technique over conventional quantization algorithms in the context of power system security analysis is the possibility of direct visualization of the system state on a graphical display, see also 8.1.2. Since this is not possible for maps of dimension higher than two, the method proposed by Bauer and Pawelzik cannot make usc of this desirable property of the Kohonen network.

Kohonen [1989], Ritter and Schulten [1988] and Obermayer [1993] observed that the coordinates of the two-dimensional lattice of the self-organizing map are organized according to those variables of the input space for which the variances are maximal, This is a very important feature for our application of the feature maps to static line overload assessment and prediction. When choosing the line loads as input vector components, the feature map organizes itself according to line loads varying between outage and overload.

A summary of the discussed results as well as further references to theoretical work is given in [Cottrell et *al.*, 1994].

5. APPLICATION OF SELF-ORGANIZING FEATURE MAPS IN POWER SYSTEMS

In technical areas Kohonen networks have successfully been applied to solve the inverse kinematic problem in robotics [Ritter et *al,* 1991]. Another promising application of self-organizing feature maps is the representation in two dimensions of high-dimensional input vectors for the design of integrated circuits [Tryba, 1992]. An application for the optimization of NP-complete combinatorial problems like the Traveling Salesperson Problem was discussed by Fort [1988].

Applications for electric power systems include static security assessment [Niebur and Germond, 1991, 1992; El-Sharkawai and Atteri, 1993], steady state stability [Mori, 1991b], fault diagnosis [Lubkeman *et* al., 19911, transformer fault diagnosis [Baumann et al,, 1991], and load forecasting, [Hsu and Yang, 1991; Germond *et* al., 1992].

5.1. Transformer fault diagnosis

Transformer fault diagnosis is a pattern recognition and classification task, which has successfully been solved with the self-organizing feature map. Obtained by impulse tests, the transformers' transadmittance (transfer function) is a good

indicator for the fault status of the transformer. The sampled magnitudes of the transadmittances of transformer faults are presented as input vectors to the Kohonen net, The phase of the transadmittance was often corrupted by noise and was therefore discarded.. An example for a discretized transfer function is shown in Fig. 12. [Baumann, T. et *al.* [1991] work with simulated training and test data.

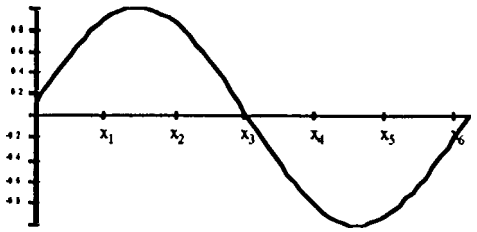

Fig. 12 Discretization of a transfer function into an input vector of dimension 6.

Fig.13 illustrates how the neurons of an 8x8 map respond to different shapes of transfer functions. Note for instance, how the weight vectors of neurons in the second row correspond slightly varying input features where the transfer function, has a pronounced maximum in class 16. The shape of the function will gradually flatten and become monotonously decreasing for class 21. Notice also, how the direct neighbors of neuron 36 all show a function whose maximum is obtained at the first discretization. Different types of transformer fauns arc classified by different neurons on the map.

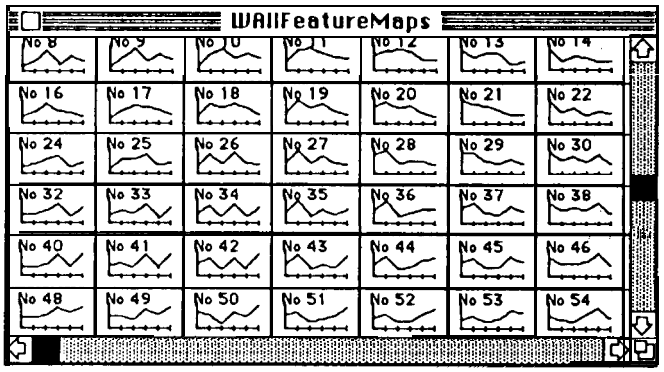

Fig.13 Organization of a 8x8 Kohonen map for transformer fault diagnosis, (adapted from [Baumann, Tschudi and Germond,. 1991]; figure provided try courtesy of the authors)

5.2. *Load forecasting*

The task of load forecasting consists of two steps, firstly the analysis of the load data with respect quality of data and with respect to different consumer behavior depending on seasons, weekdays and holidays; secondly the estimation of the load to be forecasted based on previously experienced load demands. The self-organizing feature map has successfully solved the data analysis task by creating classes of load patterns which are averages of several similar load patterns. Choosing the 24 hourly loads, next days peak load and 4 different day types as inputs to the neural net, [Germond et al., 1991] show how a 10x 10 Kohonen network maps similar load data onto neighboring classes on the map shown in Fig. 14.

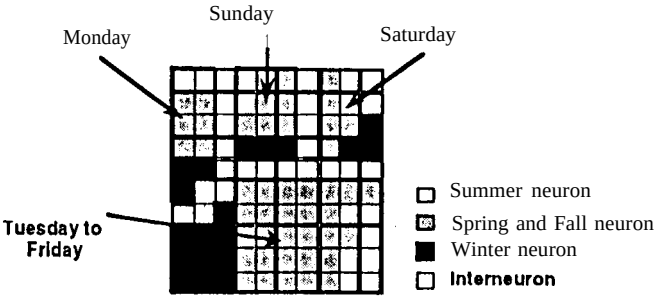

Fig. 13 Organization of a lOxlO Kohonen map for load forecasting, (adapted from [Germond *et* rd.. 1992]; figure provided by courtesy of the authors).

Notice that the pre-dominant organization of the map is due to different day types which are mapped to different clusters. Seasonal changes affect the organization inside the clusters only.

S.3. *Power System Security Assessment*

The Kohonen network for power system security analysis has been studied by Niebur and Germond, [1991, 1992]. The quantization feature of the Kohonen map is illustrated in Fig. 14. The cube shown in this figure corresponds to the safe part of the linear operating space of a 3-bus 3 line power system introduced in the previous chapter, [Niebur, 1996]. The axes of the cube are labeled by the line powers, P_{ab}, P_{ac}, P_{bc}. The surfaces of the cube are defined by the three active transfer limits of the three lines.

The self-organizing feature map quantizes the operating space into safe, critical and unsafe regions, as shown in Fig. 14. In general the security classes are not given by cubes but by a more general three-dimensional tessellation, The weight vectors of the neurons represent typical operating states which can be analyzed off-line either statistically or with conventional power system analysis tools. In the ideal case, secure and critical states are classified by the neurons in the center of the grid, and unsecure states will be classified by the neurons at the border of the grid. The inner neurons will give quite precise quantitative information on the vulnerability of the system state with respect to security limit violations. The neurons at the border will give less precise information about the insecure operating states lying far away from the secure region. However, for these inadmissible operating points, remedial action should always been taken.

The second important feature of the Kohonen map is the preservation of the topology of the input space. In other words, input vectors close in the input space should be classified by neurons close to each other on the grid.

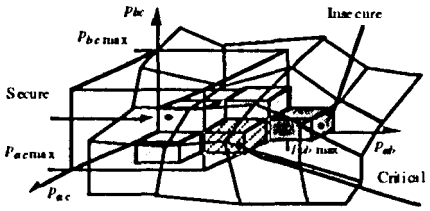

Fig. 15 Quantization of the operating space with the self-organizing feature map. The shaded cubes represent classes of operating points represented by the weight vectors of the neural net.

Since the mapping is from a high-dimensional space to a two-dimensional lattice, only an approximation of this topology preservation can be achieved, except for the case in which the input vectors lie on a two-dimensional manifold of the n-dimensional input space. Assuming the preservation is sufficiently good, the features of the classes represented by the neuron can then be displayed in two dimensions, as is demonstrated in Fig. 16.

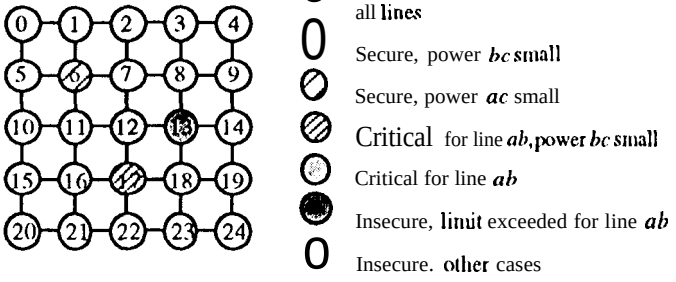

Fig.16: Feature map of security regions presented in Fig. 15.

An efficient implementation of Kohonen networks on specialized hardware for power systems static security is discussed by Cornu *et al.*, [1991].

6. CONCLLJS1ON

In summary, the Kohonen network classifies the input vectors with respect to the distance between input vector and the weight vectors of the neural net. Each weight vector therefore represents a certain number of input vectors and organizes the input space based on the probability distribution of the input data. The training algorithm quantifies the input pattern space consisting of the input vectors into at most m classes and computes the weight vector-s as representative elements of these classes. The neurons on the 2-dimensional map are organized according to those components of the input vectors having the largest variance. The map therefore provides a feature selection in addition m the data quantization.

The following properties of the self-organizing feature map are of major importance for technical application:

- Quantization of the input space.
- Robustness towards bad or missing data.
- Dirnensionality reduction of the operating vector.
- Topology preservation of the input space structure.

In addition to the presented applications to power systems, feature maps have a great potential in the area of monitoring. The two-dimensional topological representation of the operating space provides a global qualitative picture of the instantaneous situation to be monitored. This feature is not currently available in power system control centers. The stochastically evolving operating point can be monitored on a computer screen, and the direction of the trajectory of the operating point indicates whether the power system or power plant state moves out of the secure area and which constraints are most likely going to be violated.

7. ACKNOWLEDGMENT

The work described in this paper was started at the Swiss Federal Institute of Technology, Lausanne (EPFL), sponsored by EPFL, and was completed at the Jet Propulsion Laboratory, California Institute of Technology sponsored by the U.S. Department of Energy through an agreement with the National Aeronautics and Space Administration.

Reference herein to any commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

8. REFERENCES

Bauer H. U. and Pawelzik, K., "Quantifying the neighborhood preservation of self-organizing feature maps," *IEEE Trans. on Neural Networks,* vol. 3, no. 4, 1992, 570-579.

Baumann, T., Germond, A. and Tschudi, D., "Impulse test fault diagnosis on power transformers using Kohonen's self-organizing neural network," *Procs. of the Third Symposium on Expert System Application to Power Systems,* Tokyo, April 1-5, 1991, 642-647.

Cornu, T., Ienne, P., Niebur, D. and Viredaz, M., "A systolic accelerator for power system security assessment.," Accepted for publication, *Fifth International Symposium on Intelligent System Applications to Power Systems,* Montpelier, France, September 94.

Cottrell, M. and Fort, J., "Etude d'un processus d'auto-organisation," *Annrdes de l'institut Poincaré, vol. 23,* 1987, *1-20.*

Cottrell, M., Fort, J. C., Pagès, G. "Two or three things we know about the Kohonen Algorithm" *F'roes. of ESANN.* Brussels, April 1994, 235-244.

Cottrell, M. and Fort, J., "A model of retinotopy: A self-organizing process," *Biol. Cybernetics,* no. 53, 1986, 405-411

Demartines, P., "Manuels d'utilisation du simulateur de réseau de Kohonen, du simulateur de réseau de Hopfield, du simulateur de réseaux de Hérault-Jutten,". *Technical Report,* Laboratoire de Micro-electronique, EPFL, Lausanne .1990.

El-Sharkawi, M.A. and Atteri, R., "Static security assessment of power system using the Kohonen neural network, " *ANNPS93',* Tokyo, Japan, April 1993, 373-377.

Erwin, E., Obermayer, K. and Schulten, K., "Self-organizing maps: ordering, convergence properties and energy functions," Biological Cybernetics, 67, 1992a, 47-55.

Erwin, E., Obermayer, K. and Schulten, K., "Self-organizing maps: stationary states, metastability and convergence rate, " Biological *Cybernetics, 67,* 1992b, 35-45.

Fort, J. C., "Solving a combinatorial problem via self-organizing process: An application of the Kohonen algorithm to the traveling salesman problem, " *Biological Cybernetics. 59,* 1988, *33-40.*

Germond, A. J., Macabrey, N. and Baumann, T., "Application of artificial neural networks to load forecasting," *Proceedings of the* 1992 *INNS -Summer Workshop "Neural Networks Computing for the Electric Power Industry",* Stanford, CA, August 17-19, 1992. Lawrence Erlbaum Assoc., Hillsdale, N. J., 1993, 165-171.

Haykin, S., Neural *Networks, A Comprehensive Foundation,* IEEE Press #PC04036, Macmillan College Publishing Company, Inc. Englewood Cliffs, NJ, 1994.

Hertz, J., Krogh, A. and Palmer, R. G., *Introduction to the Theory of Neural Computing,* Addison Wesley, Reading, Ma, 1991.

Hochet, B., Peiris, V., Corbaz, G. and Declerq, M., "Implementation of a neuron dedicated to Kohonen maps with learning capabilities," *Procs. of the IEEE C/CC,* Boston, 1990.

Hsu, Y. -Y., Yang, C. -C., Design of Artificial Neural Networks for Short Term Load Forecasting. Part I: Self-Organizing Feature Map for Type Identification, IEE Proceedings-C, vol. 138, no. S, September 1991, pp. 407-413.

Hsu, Y. -Y., Yang, C. -C., Design of Artificial Neural Networks for Short Term Load Forecasting. Part II: Multi-Layer Feed-Forward Networks for Peak Load and Valley Forecasting, IEE Proceedings-C, vol. 138, no. 5, September 1991, pp. 414-418.

Kohonen, T., *Self-orgunizution and Associative Memory,* Third edition, Springer-Verlag, Berlin, 1989.

Kohonen, T., "Self-organized formation of topologically correct feature maps," *Biological Cybernetics,* no. *43, 59-69, 1982.*

Lehmann, C, Blayo, F. and Viredaz, M., "A generic systolic array building block for neural networks with on-chip learning, " *IEEE Trans. on Neural Networks,* vol. *4,* no. *3,* 1993, *400-407.*

Linsker, R., "Self-organization in a perceptual network. " *Computer 21, 1988, 105-117.*

Lubkeman, D., Fallen, C. and Girgis, A., "Unsupervised learning strategies for the detection and classification of transient phenomena on electric power distribution systems," *Procs. of the First International Forum on Applications of Neural Networks to Power Systems,* Seattle, WA, July 1991, 107-112.

Miller, K. D., Keller, J.B. and Stryker, M. P., "Ocular dominance column development: analysis and simulation, " Science 245, 1989, 605-615.

Miller, K. D., "A model for the development of simple cell receptive fields and the ordered arrangement of orientation columns through activity-dependent competition between on- and off-center inputs," *J. Neuroscience* 14, 1994, *409-441.*

Mori, H., Tamaru, Y, and Tsuzuki, S, "An artificial neural net based technique for power system dynamic stability with the Kohonen model, " *Procs. of the 7th on Power Industry Computer Applications Conference* , May 1991, 293-301, also in *IEEE Trans. on Power Systems,* vol. PWRS-7, no. 2, 856-864, 92105.

Niebur, D., "Introduction to concepts in artificial neural networks, " *IEEE_PES Tutorial on Artificial Neural Networks for Power Systems,* to he published, 1996.

Niebur, D. and Germond, A. J., "Power system static security assessment using the Kohonen neural network classifier, " *Procs. of the 7th Power Industry Computer Applications Conference,* Baltimore, May 1991a, also in *IEEE Transaction on Power Systems,* vol. PWRS-7, no. 2, May 1992, 865-872.

Niebur, D. and Germond, A. J., "Unsupervised neural network classification of power system static security states, " *Electrical Power and Energy Systems,* vol. 14 no. 2/3 April/June 1992, 233-242.

Niebur, D. *et al.*, "Neural network applications in power systems," *Int. Journal of Engineering Intelligent Systems, vol.* 1 no. 3, December 1993, 133-158.

Niebur, E. and Wörgötter, F., "Design principles of visual cortical columns, " *Neural Computation 6,* no. *4, 1994, 602-614*

Obermayer, K., Ritter, K. and Schulten, K., "A principle for the formation of the spatial structure of cortical feature maps," *Procs. Natl. Acad. Sci.* USA, vol. 87, 1990, 8345-8349.

Obermayer, K., *Adaptive neuronale Netze und ihre Anwendung als Modelle der Entwicklung kortikaler Karten,* Dissertationen zur künstlichen Intelligent; Bd. DISKI 24, Dr. E. Hunch (cd.), "Infix," Sankt Augustin, 1993.

Ritter, H. and Schulten, K., "Convergence properties of Kohonen's topology conserving maps: fluctuations, stability and dimension selection," Biological *Cybernetics* no. 60, 1988, 59-71.

Ritter, H. and Schulten, K., "On the stationary state of Kohonen's self-organizing mapping," *Biological Cybernetics* no. 54, 1986, 99-106.

Ritter, H., *Selbstorganisierende neuronale Karten*, Dissertation, Technische Universität München, 1988.

Ritter, H., Martinetz, T. and Schulten, K., *Neuronale Netze*, Addison Wesley, Bonn, 1991.

Ritter, 11., "Asymptotic level density for a class of vector quantization processes", *IEEE Trans. on Neural Networks, vol. 2,* no. 1, January 1991, 173-175.

Ruzicka, P., "On convergence of learning algorithm for topological maps," *Neural Network World, 4,* 1993, *413-424.*

Sirosh, J. and Miikolaienen, R., "Cooperative self-organization of afferent and lateral connections in cortical maps," *Biological Cybernetics,* in press.

Tolat, V. V., "An analysis of Kohonen's self-organizing maps using a system of energy functions," *Biological Cybernetics,* vol. *64, 1990*

Tryba V., Metzen S. and Goser K., "Designing of basic integrated circuits by self-organizing feature maps," *Procs. of Neuro-Nîmes*, 1988.

Tryba, V., *Selbstorganisierende Karten: Theorie, Anwendung und VLSI-implementierung*, Fortschritt-Berichte VDI, Reihe 9: Elektronik, Nr. 151, VDI Verlag GmbH, Düsseldorf, 1992.

Vittoz, E., Sorouchyari, E., Heim, P., Arreguit, X., and Krummenacher, F., "Analog VLSI implementation of a Kohonen map," *Procs. of the International Conference on Neural Networks,* EPF-Lausanne, October 1989.

von der Malsburg, C., "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik*, 14, 1973, *85-100.*

Willshaw, D. J., and von der Malsburg, C., "How patterned neural connections can be set up by self-organization," *Procs. R. Sot. London B 194, 1976, 431-445.*

Yang, H. and Dillon. T. S., "Convergence of self-organizing neural algorithms," *Neural Networks 5, 1992, 485-493.*

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta(t)\ \text{grad}\ E(\mathbf{w}(t))$$
$$= \mathbf{w}(t) + \eta(t) \sum_{\mu=1}^{4} \left(y^{\mu}(\mathbf{w}(t)) - y^{\mu}_{\text{target}} \right) \mathbf{x}^{\mu} \qquad (14)$$

So far wc have replaced the inversion of the pseudo-inverse by an iterative procedure. This procedure still needs the knowledge over the whole training set.

However instead of minimizing the error globally we can now try to minimize the error locall y by random] y taking one training example $(\mathbf{x}^{\mu}, y^{1}_{\text{target}})$ at a lime

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta(t) \left(y^{\mu}(\mathbf{w}(t)) - y^{\mu}_{\text{target}} \right) \mathbf{x}^{\mu}$$
$$= \mathbf{w}(t) + \eta(t)\, \delta^{\mu}\, \mathbf{x}^{\mu} \qquad (15)$$

This stochastic updating or learning rule is commonly referred to as the *LMS rule, the Widrow-Hoff rule,* or the *delta rule.* It *is one* of the earliest adaptive "neural units, called adapt *ive lines r unit* or *ADALINE* and was used for adaptive control, [Widrow and Hoff, 1960].

The iterative process converges stochastically to the minimum of the error function, if the so-called "Robbins-Monro" conditions hold for the learning rate η, [Duds and Hart, 1972]

$$\sum_{t=0}^{\infty} \eta(t) = +\infty \qquad \text{and} \qquad \sum_{t=0}^{\infty} \eta(t)^2 < \infty \qquad (16)$$

Although only applicable to linearly separable learning tasks, see remarks 7.2.3, the delta rule fulfills several of the biological paradigms. It is computationally simple, robust with respect to noisy input data as well as numerical rounding errors and it is a local adaptation scheme learning one example at a time.

It further obeys a generalization of Hebb's learning principle (8). For a fixed input and target output, the weight changes depend on two terms only, the correlation of input x and calculated output y , and a constant stimulus, the product of input and target output,

There are other supervised learning rules based on Hebb's principle like the perceptron rule and the generalized delta rule, introduced in the next tutorial chapter. Other types of neural networks trained with a different type of supervised learning like the Functional-1.ink Net are discussed in [Pao, 1989].

7,2.4. *Unsupervised learning - learning for data reduction*

In unsupervised learning the input vectors of the training set are given, but the corresponding target outputs are not specified. Unsupervised neural nets fall into the same class of tools as statistical non-parametric data analysis, clustering algorithms and encoding or decoding techniques,

Their main goal consists in data reduction. The reduction of the data set of input vectors can be achieved in two different ways: either by *reducing the dimensionality* of the input vector, or by *reducing the number* of input vectors.

The simplest neural network for unsupervised learning consists of a layer of feed-forward winner-take-all units. For each input vector only one such unit will respond, namely the unit characterized by the maximum output, respectively minimum distance, for this input vector x. The units of the network are thus competing for selection. Only the weights of the winner will be adapted. All input vectors responding to the same unit are said to forma class and the weight vector of this unit is called the class "prototype." Here, the gain function is defined to yield one for the maximum respectively minimum output, and zero otherwise,

"Winner-kWe-all" units are related to "grandmother cells" because they are responsible for selecting one specific feature, e. *g*. the feature presenting the stereotypical grandmother. Note that this representation is not robust because when one unit is removed (or one cell dies in a biological brain), all information concerning the corresponding class would be lost.

A solution to this dilemma is proposed by Kohonen's self organizing feature map where (ideally) neighboring neurons classify neighboring features and thus the loss in one neuron will result in a decrease of accuracy but not in a complete loss of information.

Let us briefly introduce the main concepts used in some types of unsupervised networks. For more detailed information see {Krogh et *al.*, 1991; Haykin, 1995]. Figs. 10, 11 and 12 show schematically how the data reduction of randomly distributed data is achieved using 3 different types of unsupervised networks.

The first unsupervised approach for the *reduction of the dimension* of the input vector falls in the class of subspace techniques where the input vector is projected on a linear subspace presenting the most salient features. Statistical principal component analysis chooses the subspace spanned by the eigenvectors of the correlation matrix of the input vector. The standard deviation of the input vectors take their maximal and minimal values along the eigenvectors corresponding to maximal and minimal eigenvalues. A simple example is shown in Fig. 10 where the data variation along the horizontal axes is more prominent than the one along the vertical axes.

A non-competitive unsupervised network for principal component analysis based on Hebb's learning rule was proposed by Oja [1989] and generalized by Sanger, [1989];

for details and references to this work see also [Haykin et al., 1995].

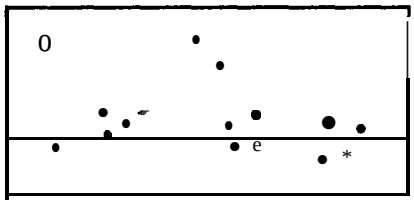

Fig. 10 Data projection onto a one-dimensional hyperplane. Each data point will be represented by its lower-dimensional projection onto the straight line. The shaded circle denotes a new input vector for which tbe projection exists, although the classification error will be large.

The second unsupervised approach for the *reduction of the number* of input vectors is based on clustering techniques. In order to reduce this number, the neural net categorizes the training vectors into classes or clusters based on the concept of similarity introduce in section 6. For the examples we will use the Euclidean distance bet ween two vectors as a measure of similarity.

In classical clustering techniques, such as the ISOdata algorithm, [Duds and Hart, 1973], clusters are formed by computing the distance between an input vector and already existing clusters. If the distance between the input vector anti the reference vector of an existing cluster is smaller than a previously defined threshold, the new input vector is grouped with this cluster; otherwise, a new cluster is formed. Functional y, a spherical neighborhood is formed around the reference vector of each ncw cl uster. Note that the diameter of the sphere is predetermined, whereas the number of clusters is not. An example of this type of clustering is presented in Fig. 11. A similar objective is achieved by the Adaptive Resonance Theory (ART) networks [Carpenter and Grossberg, 1987].

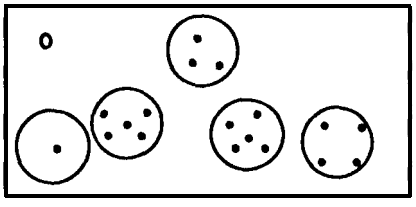

Fig. 11 Clustering of data into a variable number of classes of fixed diameter. The center' of the circles, not presented in this figure. present the class prototypes. The shaded circle denotes a new input vector which does not foil into any of the trained classes.

In vector quantization techniques based on the LBG algorithm [Linde *et al.*, 1980] or the *k-means* clustering, like the Kohonen network, [Kohonen, 1989] , the maximal number of clusters is determined by the number of neurons in the map. The weight vectors arc the reference vectors or prototypes of the class. On the other hand, the distance around the reference vector of a cluster is not predetermined and the region is, in general, not spherical. Instead, the clusters are large in the regions where the density of probability of the input vectors is small, and *vice-versa*, as shown in Fig. 12.

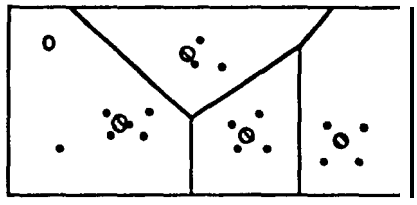

Fig. 12 Tessellation of data into a fixed number of classes of variable diameter. The striped circles, represent the class prototype. The shaded circle denotes a new input vector which does falls into one of the trained classes although the classification error will be large.

In the case of simple vector quantization, that is for a Kohonen network with winner-take-all units and no neighbor stimulation, the network minimizes the a average distortion error between the input vectors and their reference vector, The regions itself correspond (o the Voronoi tessellation, and boundaries of the regions around a cluster are hyperplanes.. More details will be presented in the chapter on Kohonen networks. Important results and references can be found in [Ritter *et al.*, 1992].

7.3 *Purpose of training in power systems*

Let us illustrate the concepts of supervised and unsupervised learning for a very simple power system shown in Fig. 13, consisting of two generation busses *a, b, one* load bus, *c* , and three lines ab, *ac, bc,* whose active power flows P_{ab}, P_{ac}, and P_{bc} *are* limited by the maximal active line powers, *i. e.* $P_{ab\,max}$, $P_{ac\,max}$ and $P_{bc\,max}$.

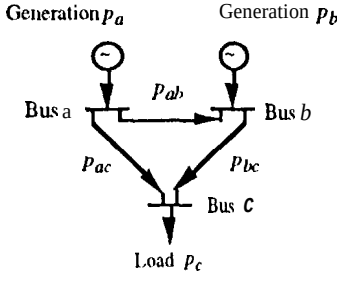

Fig. 13 A 3-bus-3-line linear power system model.

The operating vector can be chosen to consist of the active line powers (p_{ab}, *pat*, P_{bc}) In this case the secure operating space is defined by a parallelepiped whose boundaries are determined by $P_{ab\,max}$, $P_{ac\,max}$ and $P_{bc\,max}$, see Fig. 14. For simplicity we will throughout this work refer to this parallelepiped as the security "cube". Operating points inside the shaded cube are secure, points inside but at the border are critical and operating point outside the shaded cube are unsecure because they violate at least one constraint on the maximum admissible line powers,

This example is based on several simplifications. Only active powers have been considered, In the general case the cube has

to be replaced by a non-linear manifold. Furthermore, not all vectors of the three-dimensional power system operating space shown in Fig. 14 represent feasible operating states, since Voltage-VAr constraints and Kirchhoff's laws apply for each bus and each line. Nevertheless, the example illustrates conveniently the differences between supervised and unsupervised learning.

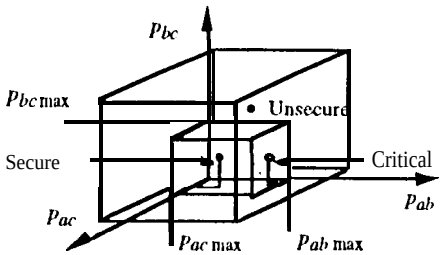

Fig. 14 The operating space of the 3-bus-3-line. linear power system model,

Supervised training approximates the boundaries of the operating space for the training set and interpolates in between known data points, It basically constructs separating hyperplanes (manifolds in the non-] inear case) corresponding to the surfaces of the shaded secure cube in Fig. 14. An example for this technique as well as several enhancements are discussed in [El-Sharkawi *et* al., 1991].

However, because in the general case the dimension of the operating space is very high (in the order of 500 for a medium sized power system at the transmission level), it is not feasible to generate a set of operating points which is densely distributed in the operating space and to analyze the operating points with multiple contingency analysis off-line. In order to overcome this "curse of dimensionality", *unsupervised learning* tackles the dimensionality problem first based on two different approaches

a) Subspace techniques
b) Quantization techniques.

The simplest *subspace technique is* the conventional contingency ranking techniques. If for example the outage of line ah is selected as the most important contingency, the operating space of the linear model is projected to a two-dimensional subspace as illustrated in Fig. 15.

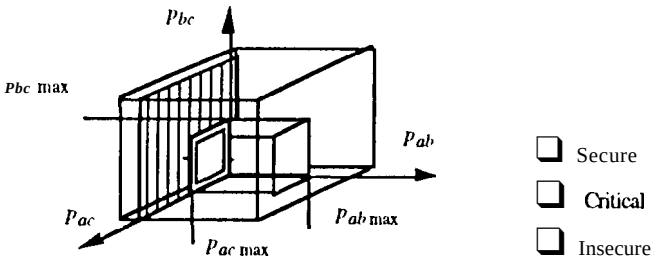

Fig. 15 Limitation uf the number of contingencies.

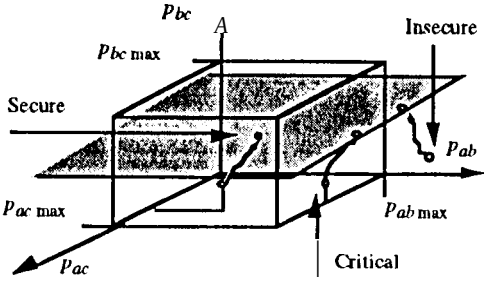

Fig. 16 Reduction of the dimension of the operating vector.

Conventional load flow analysis examines the projection of the base case onto this subspace. Supervised techniques are also applied to construct the boundaries of the projected reduced, security cube, see [El-Sharkawi *et al.,* 1991].

Fig. 16 shows an more general example of the reduction of the operating space by a lower-dimensional manifold, Depending on the projection used for reduction, the manifold may be a linear or even orhtogonal subspace.

In [Weerasooriya and El-Sharkawi, 1991] the principal component analysis method (also called Karhunen-Lobe expansion) is used to reduce the dimensionality of the training vectors and construct the eigenspace corresponding to the most significant components of the input vector. The researchers implemented their approach in a conventional algorithmic manner instead of using Oja's and Sanger's neural net approach.The second class of unsupervised approaches encountered in power system security assessment are *quantization techniques.* Fig. 17 shows an example of the quantization of the operating space into classes of typical states. Depending on the distance measure used for classification, classes may be hyperboles, spheres or in the case of the self-organizing feature map, of a more general form because of the arrangement of neurons on a grid. The classes usually do not divide the cube crisply in secure and insecure areas, but may contain critically high loaded as well as slightly overloaded cases.

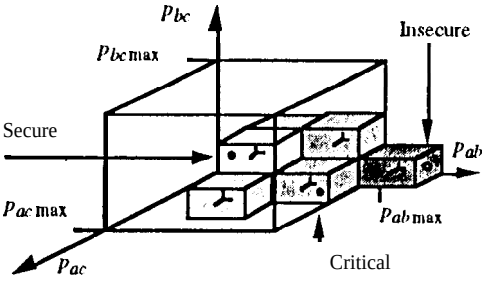

Figure 17: Quantization of the operating space.

The two different clustering approaches discussed in section 7.3 have been applied to security assessment.

In the case of a small space station transmission system, Sobajic et *al. [1990]* quantize the operating space into a variable number of hyperspheres of fixed radius using an unsupervised ART2-like ANN algorithm.

In [Niebur and Germond, 1991] Kohonen's self-organizing feature map is used for the quantization of the operating space, The maximal number of classes is given by the number of neurons whose weight vector represents typical operating states. The size of each class depends on the density of the probability distribution of the training vectors. The operating space is represented on the two-dimensional feature map by secure and insecure regions. This case will be discussed in more detail in the following chapter.

7.4 *Comparison of supervised and unsupervised learning*

Although usually discussed on equal terms, there is an important difference between supervised and unsupervised learning. Unsupervised learning helps to organize complex features into classes whereas supervised learning will then calculate follow-up features for specific classes.

Unsupervised networks can therefore be viewed as a data preprocessing step which reduces the size of the data set before learning the data's characteristics with supervised learning. The *Functional Link Net* (FLN) is often used in combination with the ART2 network [Sobajic and Pao, 1988]. Other ANNs combining an unsupervised and a supervised step are the *Counter-Propagation Network* (CPN) [Hecht-Nielsen, 1988], and the *Radial Basis Functions Net work* (RBF) [Moody and Darken, 1989]. The CPN combines a Kohonen map layer with a feed forward layer. In the case of the RBF, clustering can bc achieved by any unsupervised learning or the k-means algorithm, and the neurons of the hidden layer arc represented by these means. The architecture of the supervised part is a linear feed-forward layer. In contrast to the winucr-take all scheme in the. Kohonen network, Gaussian activation functions stimulate several neurons at the same time and the output of the network is a weighted sum of these activations.

For security assessment, the combination of an unsupervised step for operat ing space reduction and a supervised step for operating state classification has been applied by several researchers including [Sobajic and Pao, 1990; El-Sharkawi *et* al., 1991; Ranaweera and Karady, 1994]

Another example in power systems, where supervised and unsupervised networks arc employed for data clustering and estimation is the area of load forecasting [Hsu and Yang 199 1]. A Kohonen network separates the forecasting data into representative classes like summer, winter, autumn and spring and further into weekdays and holidays (see also [Macabrey et *al.* 1991]. For each class of data a supervised network is then used for load prediction for the classes data points. For a similar purpose Ranaweera *et al.* [199S] apply the RBF network in the area of load forecasting. Further detailed examples will be discussed in the other tutorial chapters.

8. SUMMARY

We have presented an overview over different types of neural units characterized by their input, output, weight vector, gain function, architecture, processing and learning algorithm.

Tables II-V give a short overview on the different characteristics of neural networks and anon-exhaustive list of examples.

TAB LE II

Neural net parameters	
Input vector x	Number of neurons
Output vector y	Gain function g(h)
Weight vector w	Learning rate $\eta(t)$

TABLE III

Architecture	Examples
Layered	**Multi-layer perception**
Fully connected	**Hopfield**
Lateral connections	**Kohonen**
Hybrid networks	Radial Basis Functions net
	Counter-Propagation net
	Boltzmann machine

TABLE IV

Processing (x, w given, calculate y)	Examples
Feed-forward,	Adaline
feed x once to get y	Multi-layer perceptron
	Kohonen
Recurrent, iterate x to get y	Hopfield
	Diagonally recurrent ANN

TABLE V

Training (x given, calculate w)	Examples
Supervised learning	Delta rule
(y given)	Back-propagation
Unsupervised learning	Principal Component Analysis
(no y given)	Self-organization

9. ACKNOWLEDGMENT

The work described in this paper was started at the Swiss Federal Institute of Technology, Lausanne (EPFL), sponsored by EPFL, and was completed at the Jet Propulsion Laboratory, California Institute of Technology sponsored by the U.S. Department of Energy through an agreement with the National Aeronautics and Space Administration.

Reference herein to any commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

10. REFERENCES

Cajal, S. R., "La fine structure des centres nerveux," *I'roes. Roy. Sot.* Lend. 55, 1894, 444-468.

Carpenter, G. A. and Grossberg, S., "ART2: Self-organization of stable category recognition category of analog input patterns," *Applied Optics 26, 1987, 4919-49.*

Churchland, P. S. and Sejnowski, T. J., *The Computational Brain, The* MIT Press, Cambridge MA, 1992.

Cornu, T., Ienne, P., Niebur, D. and Viredaz, M., "A systolic accelerator for power system security assessment., " *Fifth International Symposium on Intelligent System Applications to Power Systems,* Montpelier, France, September 94, 431-438.

Duda, R. O. and Hart, P. E., *Pattern* Classification *and Scene Analysis,* John Wiley and Sons, Inc., New York, 1973.

El-Sharkawi, M. A. et al., "Artificial neural networks as operating aid for an on-line static security assessment of power system s," *Procs. of the Tenth Power System Computation Conference.* Graz, 1990, 895-901.

El-Sharkawi, M. A. et al., "Neural networks and their application to power engineering, " *Control and* Dynamic *Systems* 41, Academic Press, 1991.

Haykin, S., *Neural Networks, A Comprehensive Foundation,* IEEE Press #PC04036, Macmillan College Publishing Company, Inc. Englewood Cliffs, NJ, 1994.

Hertz, J., Krogh, A. and Palmer, R. G., *Introduction to the Theory of Neural Computing,* Addison Wesley, Reading, Ma, 1991.

Hopfield, J. J., "Neural networks and physical systems with emergent collective computational abilities, " *Procs. Nut. Acad. Sc.*, 79, 1982, 2554-2558,

Hsu, Y. -Y., Yang, C. -C., Design of Artificial Neural Networks for Short Term Load Forecasting. Part I: Self-Organizing Feature Map for Type Identification, Part II: Multi-Layer Feed-Forward Networks for Peak Load and Valley Forecasting, *IEE Proceedings-C, vol.* 138, no. 5, September 1991, pp. 407-418.

Kohonen, T., *Self-organization und Associative Memory,* I'bird edition. Springer-Verlag, Berlin, 1989.

Linde, Y., Buzo, A> and Gray, R. M., "An algorithm for vector quantizer design," *IEEE Transactions on Communications,* vol. COM-28, 1980, 84-95.

McCulloch, W. S. and Pitts, W. 11., "A logical calculus of the ideas immanent in nervous activity, ", " *Bulletin of Mathematical Biophysics, vol. 5, 1943,* 115-133.

Mead, C., Analog VLSI and Neural Systems, *Addison- Wesley,* Reading, MA, 1989.

Moody, J. and Darken, C., "Fast learning in networks of locally tuned processing units," Neural Computation 1, 1989, 281-294.

Niebur, D., "An Example of unsupervised networks - Kohonen's self-organizing feature map," *IEEE_PES Tutorial on Artificial Neural Networks for Power Systems,* to be published, 1996.

Niebur, D. and Germond, A. J., "Power system static security assessment using the Kohonen neural network classifier," *Procs. of the 7th Power Industry Computer Applications Conference,* Baltimore, May 1991a, also in *IEEE Transaction on Power Systems,* vol. PWRS-7, no. 2, May 1992, 865-872.

Niebur, D. *et al.,* "Neural network applications in power systems," *Int. Journal of* Engineering *Intelligent Systems, vol.* 1 no, 3, December 1993, 133-158.

Oja, E., "Neural networks, principal components and subspaces, " *Int. J. of Neural Systems,* vol. 1, no. 1, 1989, 61-68.

Pao, Y. H., *Adaptive Pattern Recognition and Neural Networks,* Addison-Wesley, Reading, MA, 1989.

Ranaweera, D. K. and Karady, G., "Power system security analysis using radial basis function neural network, " *Procs. of the 4th Symposium on Expert System Application* to *Power Systems,* Melbourne, 1993, 272-274.

Ranaweera, D. K., Hubele, N., and Papalexopoulos, A. "Applications of radial basis function neural network model for short-term load forecasting", *IEE Proc. -Generation, Transmission and Distribution,* vol. 142, no. 1, January 1995, 45-50.

Ritter, H., Martinetz, T. and Schulten, K., *Neural Networks,* Addison Wesley, 1992.

Rumelhart, D. E., Hinton, G. E. and Williams, R. J., "Lerrming internal representations by error propagation, " in Rumelhart, D. E. and McClelland, J., L. (eds.), *Parallel distributed processing: Explorations in the Microstructure of Cognition,* 1, 318-362, MIT Press, Cambridge, MA, 1986.

Sobajic, D. J., Pao, Y.-H. and Dolce, J., "Real-time security monitoring of electric power systems using parallel associative memories," IEEE *International Symposium on Circuit sand Systems, New* Orleans, Louisiana, May 1-3, 1990, 2929-2932.

Weerasooriya, S. and El-Sharkawi, M. A., "Use of Karhunen-Loéve expansion in training neural network for static security assessment, " *First International Forum on Applications of Neural Networks to Power Systems,* Seattle, WA, July *23-26, 1991, 59-64.*

Werbos, P. J., Beyond regression: New tools for prediction and analysis **in the behavioral sciences,** *Doctoral Dissertation,* Al. Math., Harvard University, November 1974.

Widrow, B. and Hoff, M. E., "Adaptive switching circuits," *IRE WESCOMN Convention Record, 1960, 96-104.*